

每周工作汇报

姓名	侯宇轩	开始日期	2018.9.17	结束日期	2018.9.23
----	-----	------	-----------	------	-----------

1. 本周任务与计划

1.1 研究任务

阅读蔡老师新布置的论文：PDE-Net: Learning PDEs from Data，学习其中的方法，思考如何用其对 level-set 进行改进，来应用在神经纤维瘤分割上。

2. 本周工作概要

2.1 当前的进展

论文的报告如下：

PDE-Net: Learning PDEs from Data
Zichao Long 等
北京大学
ICML 2018

摘要

在深入学习神经网络设计的最新发展的启发下，我们提出了一种新的前馈深度网络，称为 PDE-net，同时实现两个目标：准确预测复杂系统的动态(dynamics)，揭示隐含的 PDE 模型。与现有方法相比，该方法具有最大的灵活性，因为它同时学习 PDE 模型中的微分算子和非线性响应方程(response function)。

该 PDE 网络的特点是所有 filter 都经过了适当的约束，使我们能够容易地识别主要的 PDE 模型，同时仍然保持网络的表达力(expressive)和预测力(predictive)。这些约束是通过充分利用微分算子的阶数与 filters 的和规则的阶数(the orders of sum rules of filters)（源自小波理论的一个重要概念）两者之间的关系设计的。数值实验显示 PDE-net 有可能发现所观察到的动

态中隐藏的 PDE，并在相当长的时间内预测其动态，即使在嘈杂的环境中。

方法

在本文中，我们设计了一个深度前馈网络，基于以下常规的非线性演化 PDE：

$$u_t = F(x, u, \nabla u, \nabla^2 u, \dots), \quad x \in \Omega \subset \mathbb{R}^2, \quad t \in [0, T].$$

PDE-net 的目标是学习非线性响应函数 F 的形式(form)，并进行精确的预测。与现有工作不同，该网络只需要少量的关于 F 形式的知识，同时不需要与其相关的微分算子的知识（除了它们的最大可能阶数）及其相关的离散逼近。函数 F 可以使用神经网络或其他机器学习方法学习，而微分算子的离散近似则使用卷积核(即滤波器)结合响应函数 F 的学习来学习。如果我们对响应函数 F 的形式有先验知识，就可以利用附加信息轻松地调整网络结果。这可以简化训练并提高结果。

我们的方法的一个新颖之处是，我们对可学习的 filter 施加适当的约束，因此可以在识别主要(governing)PDE 模型的同时保持 PDE-net 的表现力和预测力。这一点使我们的方法不同于现有的深度卷积网络，因为现有的网络需要下面的假设：响应函数(response function)的形式是已知的，或对微分算子有固定的逼近(fixed approximation)。换言之，我们提出的方法不仅可以对动力学(dynamics)进行逼近，准确地预测其未来的行为，也能够显示驱使动力学的隐藏方程(hidden equations)。

PDE-Net

给出一系列对某个物理量的度量 $\{u(t_i): t = t_0, t_1, \dots\}$ ，限制在定义域 $\Omega \in \mathbb{R}^2$ 中。

$u(t_i): \Omega \rightarrow \mathbb{R}$ 。我们想要找到数据的主要(governing)PDE。假设观察的数据与以下形式的 PDE 有关：

$$u_t(t, x, y) = F(x, y, u, u_x, u_y, u_{xx}, u_{xy}, u_{yy}, \dots), \quad (1)$$

其中， $(x, y) \in \Omega \subset \mathbb{R}^2, t \in [0, T]$ 。我们的目标是设计一个前馈网络对 PDE 进行近似，满足如下条件：

- 1) 我们可以预测方程在任意长时间内的动力学行为。
- 2) 我们能够揭示响应函数 F 的形式，还有与此相关的微分算子。

PDE-net 有两个主要组成部分：

- 1) 对 PDE 中的微分算子中的自动识别，同时给出离散近似估计(discrete approximations)

2) 近似估计响应函数 F 。
这两部分使用同一网络实现。

卷积与微分

Cai et al. (2012); Dong et al.(2017)给出了卷积与微分的深层关系。他们在论文中讨论了 filter 的和规则的阶(order of sum rules) 与微分算子的阶的关系。注意, 在小波理论中, 和规则的阶与消失动量(vanishing moments)的阶紧密相关(Daubechies, 1992; Mallat, 1999).

定义 2.1(和规则的阶)

对于一个 filter q , 我们说 q 有和规则的阶 $\alpha = (\alpha_1, \alpha_2), \alpha \in \mathbb{Z}_+^2$, 当

$$\sum_{k \in \mathbb{Z}^2} k^\beta q[k] = 0 \quad (2)$$

对于所有的 $\beta = (\beta_1, \beta_2) \in \mathbb{Z}_+^2, |\beta| := \beta_1 + \beta_2 < |\alpha|$ 成立, 对于所有的 $\beta \in \mathbb{Z}_+^2, |\beta| = |\alpha|, \beta \neq \alpha$ 也成立。

若 (2) 对于一切 $\beta \in \mathbb{Z}_+^2, |\beta| < K$ 成立, 除了对一些 $\beta \neq \bar{\beta}, \bar{\beta} \in \mathbb{Z}_+^2, |\bar{\beta}| = J < K$ 的情况, 那

么我们称 q 有全和规则(total sum rules), 阶为 $K \setminus \{J + 1\}$

实践中, filter 一般是有限的, 可以被看作矩阵。对于一个 $N \times N$ 的 filter q (N 为奇数),

假设 q 的序号从 $-\frac{N-1}{2}$ 开始, 那么 (2) 可以被写作如下形式:

$$\sum_{l=-\frac{N-1}{2}}^{\frac{N-1}{2}} \sum_{m=-\frac{N-1}{2}}^{\frac{N-1}{2}} l^{\beta_1} m^{\beta_2} q[l, m] = 0.$$

推论 2.1

Dong et al. (2017)使用以下推论将和规则的阶与微分算子的阶结合起来。

令 q 为一个 filter，带有和规则的阶 $\alpha \in \mathbb{Z}_+^2$ 。那么对于一个 \mathbb{R}^2 上的光滑函数 $F(x)$ ，有

$$\frac{1}{\varepsilon^{|\alpha|}} \sum_{k \in \mathbb{Z}^2} q[k] F(x + \varepsilon k) = C_\alpha \frac{\partial^\alpha}{\partial x^\alpha} F(x) + O(\varepsilon), \text{ as } \varepsilon \rightarrow 0. \quad (3)$$

此外，若 q 对于某些 $K > |\alpha|$ 有全和规则的阶 $K \setminus \{|\alpha| + 1\}$ ，那么

$$\frac{1}{\varepsilon^{|\alpha|}} \sum_{k \in \mathbb{Z}^2} q[k] F(x + \varepsilon k) = C_\alpha \frac{\partial^\alpha}{\partial x^\alpha} F(x) + O(\varepsilon^{K-|\alpha|}), \text{ as } \varepsilon \rightarrow 0. \quad (4)$$

根据推论 2.1，一个 α 阶的微分算子可以被一个具有和规则阶 α 的卷积 filter 所近似。同

时，根据（4）式，若该 filter 有全和规则的阶 $K > |\alpha| + k, k \geq 1$ ，那么可以形成对微分算子的高阶近似。

例如，考虑一个 filter $q = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}$ 。

它含有和规则的阶 $(1,0)$ ，一个全和规则的阶 $3 \setminus \{2\}$ 。根据推论，加上一个常数和适合的缩

放， q 可以对离散化的 $\frac{\partial}{\partial x}$ 进行二阶近似。

$$q = \begin{pmatrix} -1 & 0 & 1 \\ 1 & 0 & -1 \\ 0 & 2 & 0 \\ 0 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}$$

$$\sum_{l=-1}^1 \sum_{m=-1}^1 q[l,m] = 0$$

$$\Rightarrow (-1)^{p_1} (-1)^{p_2} + (-1)^{p_1} (-1)^{p_2} q[-1,1] + (0)^{p_1} (-1)^{p_2} q[1,-1] + (1)^{p_1} (-1)^{p_2} q[1,1]$$

$$= (-1)^{p_1+p_2} \cdot 1 + (-1)^{p_1} \cdot (-1) + (-1)^{p_1} \cdot 1 + (-1)$$

$$= (-1)^{p_1+p_2} + (-1)^{p_1+1} + (-1)^{p_1-1} + (-1)^{p_1+1}$$

$$|p| = 0 \quad \begin{cases} ① p_1=0, p_2=0 \Rightarrow 1 + (-1) + 1 + (-1) = 0 \checkmark \\ ② p_1=1, p_2=0 \Rightarrow -1 + 1 + 1 + (-1) = 0 \checkmark \\ ③ p_1=0, p_2=1 \Rightarrow 1 + (-1) + (-1) + 1 = 0 \checkmark \end{cases}$$

$$|p| = 1 \quad \begin{cases} ④ p_1=2, p_2=0 \Rightarrow 1 + (-1) + (-1) + 1 = 0 \checkmark \\ ⑤ p_1=1, p_2=1 \Rightarrow 1 + 1 - 1 - 1 = 0 \checkmark \\ ⑥ p_1=0, p_2=2 \Rightarrow 1 - 1 + 1 - 1 = 0 \checkmark \end{cases}$$

$$|p| = 2 \quad \begin{cases} ⑦ p_1=3, p_2=0 \Rightarrow \checkmark \\ ⑧ p_1=2, p_2=1 \Rightarrow -1 - 1 - 1 - 1 = -4 \times \\ ⑨ p_1=1, p_2=2 \Rightarrow \checkmark \\ ⑩ p_1=0, p_2=3 \Rightarrow \times \end{cases}$$

$$j=2, K=3, j=1 \quad \text{全排列 } 3! = 6 \Rightarrow 3! = 6 \Rightarrow 3! = 6$$

moment 矩阵(moment matrix)

下面介绍 moment 矩阵，用于 PDE-net 中对 filter 进行限制。对于一个 $N \times N$ filter q ，定义 q 的 moment 矩阵

$$M(q) = (m_{i,j})_{N \times N}, \quad (5)$$

$$m_{ij} = \frac{1}{(i-1)!(j-1)!} \sum_{k_1 \in \mathbb{Z}^2} k_1^{i-1} k_2^{j-1} q[k_1, k_2] \quad \text{其中 } i, j = 1, 2, \dots, N.$$

为了简便，我们可以称

$M(q)$ 的第 (i, j) 号元素为 q 的 $(i-1, j-1)$ -moment。将 (5) 与推论 2.1 结合，我们容易发现，应用了 $M(q)$ 的规则之后，filter q 可以被设计用来对任意微分算子进行任意阶的近似。

例如，我们想要对 $\frac{\partial u}{\partial x}$ （加上一个常数）使用卷积 $q \odot u$ 来近似， q 为一个 3×3 的 filter。

那么我们在 $M(q)$ 上考虑如下的约束：

$$\begin{pmatrix} 0 & 0 & \star \\ 1 & \star & \star \\ \star & \star & \star \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & \star \\ 0 & \star & \star \end{pmatrix}. \quad (6)$$

这里 \star 表示对对应位置没有约束。(6) 式中左边矩阵给出的约束可以保证至少一阶近似，

而右边矩阵给出的约束可以保证至少二阶近似。特别地，若 $M(q)$ 的每一项均有约束，例如

$$M(q) = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$
，对应的 filter 可以被独一无二的确定。这种情况，我们称之为冻结(frozen) filter。

值得注意的是，filter 的近似能力是被其大小所限制的。一般来说，更大的 filter 可以用于近似更高阶的微分算子，或者在较低阶的微分算子有更高阶的近似；然而，更大的 filter 也带来的更大的内存和计算开销。

网络结构

$$u_t(t, x, y) = F(x, y, u, u_x, u_y, u_{xx}, u_{xy}, u_{yy}, \dots), \quad (1)$$

对于（1）所给出的 PDE，我们使用向前欧拉法来将时间离散化。

一. 前向欧拉法

当两相邻离散点之间的间隔较小时，用一阶差商取代一阶导数

$$\frac{y(t_{k+1}) - y(t_k)}{t_{k+1} - t_k} \approx y'(t_k)$$

令步长 $h = t_{k+1} - t_k$ ，则

$$y(t_{k+1}) \approx y(t_k) + y'(t_k)h$$

其近似值为：

$$y_{k+1} = y_k + y'_k h$$

单个 $\delta_t - Block$

令 $\hat{u}(t_{i+1}, \dots)$ 为根据 t_i 时刻的 u 预测的 t_{i+1} 时刻的函数值。那么有

$$\tilde{u}(t_{i+1}, \cdot) = D_0 u(t_i, \cdot) + \Delta t \cdot F(x, y, D_{00}u, D_{10}u, D_{01}u, D_{20}u, \dots). \quad (7)$$

这里，算子 D_0 、 D_{ij} 是卷积算子，他们的 filter 由 q_0 、 q_{ij} 确定，即：

$D_0 u = q_0 \odot u$, $D_{ij} u = q_{ij} \odot u$. 算子 D_{01}, D_{11} 等对微分算子进行了近似，即： $D_{ij} u \approx \frac{\partial^{i+j} u}{\partial x^i \partial y^j}$. 算

子 D_0 、 D_{00} 是特殊的均值算子；使用这些均值算子而不是恒等映射的原因是这样可以增强网络的表达能力，使其可以捕捉到更复杂的运动。

我们并未假设观察到的运动是由形式（1）的 PDE 所支配的；与此相反，我们假设 PDE 的最高阶小于某个正整数。那么，对 F 的近似就相当于解决一个多变量回归问题，该问题可

以被点对点(point-wise)神经网络（在计算域 Ω 中共享权重）或者其他传统机器学习方法近似。

将对微分算子和非线性函数的估计结合，我们得到一个对（7）的估计框架，成为一个

$\delta_t - Block$ 。

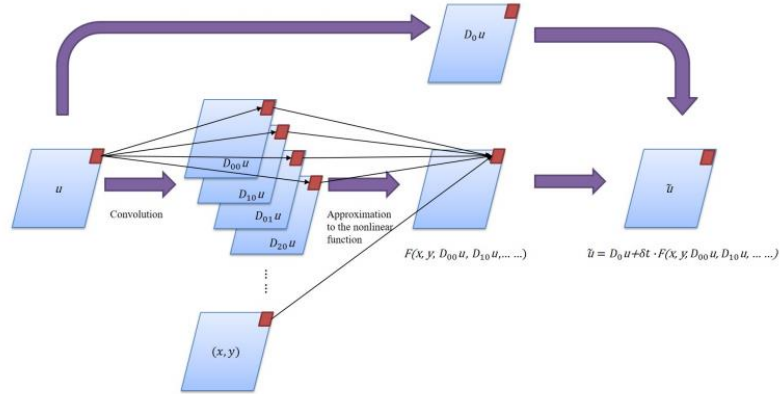


Figure 1. The schematic diagram of a δt -block.

PDE-NET（多个 $\delta_t - Blocks$ ）

单个 $\delta_t - Block$ 只能保证对一步动作预测精度，这并没有将累计误差考虑进来。为了实

现长期预测，我们将多个 $\delta_t - \text{Blocks}$ 堆叠成一个深度网络，称之为 PDE-net。Section 3 将会介绍堆叠多个 blocks 的重要性。

PDE-net 可以被简单地如下描述：

- (1) 将一个 $\delta_t - \text{Block}$ 堆叠多次；
- (2) 在所有的 $\delta_t - \text{Blocks}$ 中共享参数。

给定一个输入 $u(t_i, \cdot)$ ，训练一个含有 n 个 $\delta_t - \text{Blocks}$ 的 PDE-net 需要将累计误差

$\|u(t_{i+n}, \cdot) - \tilde{u}(t_{i+n}, \cdot)\|_2^2$ 最小化。 $\tilde{u}(t_{i+n}, \cdot)$ 是经过 PDE-net (n 个 $\delta_t - \text{Blocks}$) 后的输出。

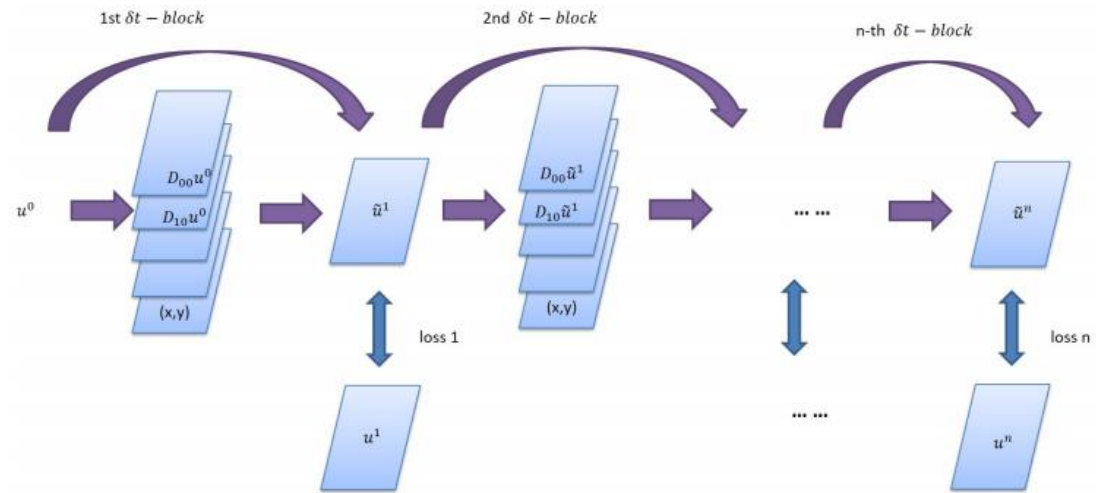


Figure 2. The schematic diagram of the PDE-Net.

损失函数和限制条件(constraints)

考虑数据集 $\{u_j(t_i, \cdot) : i, j = 0, 1, \dots\}$, j 代表未知运动在某种初始状态下的第 j 个 solution

path。我们将训练含有 n 个 $\delta_t - \text{Blocks}$ 的 PDE-net。给定 $n \geq 1$, 每一对数据 $\{u_j(t_i, \cdot), u_j(t_{i+n}, \cdot)\}$

是一个训练样本。我们选用简单的 L2-loss 进行训练：

$$L = \sum_{i,j} l_{ij}, \text{ where } l_{ij} = \|u_j(t_{i+n}, \cdot) - \tilde{u}_j(t_{i+n}, \cdot)\|_2^2,$$

其中 $\tilde{u}_j(t_{i+n}, \cdot)$ 是以 $u_j(t_i, \cdot)$ 为输入的 PDE-net 的输出。

所有在 PDE-net 中的 filters 都用它们的 moment 矩阵进行了适当的限制。令 q_0 和 q_{ij} 为 D_0

和 D_{ij} 表示的 filter。我们利用下列限制

$$(M(q_0))_{1,1} = 1, (M(q_{00}))_{1,1} = 1$$

对 $i + j > 0$, 有

$$(M(q_{ij}))_{k_1, k_2} = 0, k_1 + k_2 \leq i + j + 2, (k_1, k_2) \neq (i + 1, j + 1)$$

$$(M(q_{ij}))_{i+1, j+1} = 1.$$

为了显示可训练 filter 的重要性，我们将使用上述限制的 PDE-net 和完全确定的“冻结”PDE-net 进行对比。

初始化与训练

在 PDE-net 中，参数可以被划分为 3 组：1) 用于近似微分算子的 filter；2) 点对点神经网络（用于估计函数 F ）中的参数；3) 超参数，例如 filters 的大小、数量等等。点对点神经网络

中的参数在计算域 Ω 中是共享的，并使用随机高斯采样作为初始化。对于 filters，我们将他们“冻结”至他们对应的微分算子。

我们并没有直接训练 n 层的 PDE-net。与此相反，我们采用了逐层训练(layer-wise training)，这样可以提高训练速度（详情见(Long et al., 2018)）。在每个 δ_t - Block 中的参数在

层间被共享。此外，我们在训练第一个 δ_t - *Block*前增加了一个预热(warm-up)步骤。该步骤是为了给点对点神经网络总的参数一个良好的初始猜测值；该步骤使用冻结的 filter 来完成。

数值研究：对流-扩散方程

对流扩散方程是经典的 PDE，用于描述粒子、能量等在一个物理系统中的对流和扩散。

模拟数据，训练与测试

我们考虑一个 2 维线性变量-系数传播-扩散系统方程 (linear variable-coefficient convection-diffusion equation)，在 $\Omega = [0, 2\pi] \times [0, 2\pi]$ 上。

$$\begin{cases} \frac{\partial u}{\partial t} &= a(x, y)u_x + b(x, y)u_y + 0.2u_{xx} + 0.3u_{yy}, \\ u|_{t=0} &= u_0(x, y), \end{cases} \quad (8)$$

其中， $(t, x, y) \in [0, 0.3] \times \Omega$

$$\begin{aligned} a(x, y) &= 0.5(\cos(y) + x(2\pi - x)\sin(x)) + 0.6, \\ b(x, y) &= 2(\cos(y) + \sin(x)) + 0.8. \end{aligned}$$

数据是使用高精度数值方法（在 50x50 网格上离散化，时间步 $\delta_t = 0.015$ ）求解问题（8）

生成的。我们假设有循环边界条件，初始值 $u_0(x, y)$ 生成于

$$u_0(x, y) = \sum_{|k|, |l| \leq N} \lambda_{k,l} \cos(kx + ly) + \gamma_{k,l} \sin(kx + ly), \quad (9)$$

其中 $N = 9$, $\lambda_{k,l}, \gamma_{k,l} \sim \mathcal{N}\left(0, \frac{1}{50}\right)$, k, l 是随机选取的。我们同时在数据中加入了噪声：

$$\hat{u}(x, y, t) = u(x, y, t) + 0.015 \times MW \quad (10)$$

其中 $\mathbf{M} = \max_{x,y,t}\{u(x,y,t)\}$, $W \sim \mathcal{N}(0,1)$, $\mathcal{N}(0,1)$ 表示标准正态分布。

假设我们有先验知识：所求 PDE 是线性的，并且阶数不大于 4。那么响应函数 F 有以下形式：

$$F = \sum_{0 \leq i+j \leq 4} f_{ij}(x,y) \frac{\partial^{i+j} u}{\partial x^i \partial y^j}.$$

PDE-net 中的每一个 δ_t -block 可以写作

$$\begin{aligned} \tilde{u}(t_{n+1}, \cdot) = & D_0 u(t_n, \cdot) \\ & + \delta t \cdot (c_{00} D_{00} u + c_{10} D_{10} u + \dots + c_{04} D_{04} u), \end{aligned}$$

其中 $\{D_0, D_{ij}; i+j \leq 4\}$ 是卷积算子， $c_{ij}; i+j \leq 4$ 是二维数组，用于在 Ω 上拟合函数

$f_{ij}(x,y)$ 。拟合是通过采用对每一部分进行二次多项式插值(piecewise quadratic polynomial interpolation) 同时在每一部分的边界进行光滑变换(smooth transition)完成的。与卷积算子 $\{D_0, D_{ij}; i+j \leq 4\}$ 相关的 filters，还有每一部分二次多项式的系数都是网络中的可训练参数。

在训练与测试中，数据是在线生成的；filter 的大小设为 5x5 或者 7x7；每一个 δ_t -Block 中的可训练参数总数约为 17k。在训练中，我们使用 LBFGS 而不是传统的 SGD 来优化。我们构造的 PDE-net 至多 20 层，在每一层（即， δ_t -Block）中输入 28 个样本，那么需要在每一个 batch 中输入 560 个样本。

结果与讨论

预测长期动态

在训练一个含有 n 个 δ_t - *Blocks* 的 PDE-net 后，我们随机生成 560 个初始猜测（根据式子 (9), (10)），输入 PDE-net，并在预测的动态（即，PDE-net 的输出）和实际的动态（即，由 (8) 的数值方法求出的结果）之间求正则化的误差。在真实数据 \mathbf{u} 和预测数据 $\tilde{\mathbf{u}}$ 之间的正则化误差定义是 $\epsilon = \frac{\|\tilde{\mathbf{u}} - \mathbf{u}\|^2}{\|\mathbf{u} - \bar{\mathbf{u}}\|^2}$ ，其中 $\bar{\mathbf{u}}$ 是 \mathbf{u} 的空间均值。figure 3 显示了误差图，figure 4 显示了一些预测的动态。

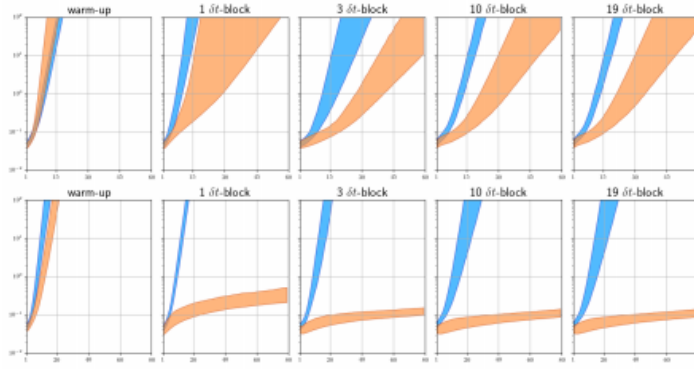


Figure 3. Prediction errors of the PDE-Net (orange) and Frozen-PDE-Net (blue) with 5×5 (first row) and 7×7 (second row) filters. In each plot, the horizontal axis indicates the time of prediction in the interval $(0, 60 \times \delta t] = (0, 0.9]$, and the vertical axis shows the normalized errors. The banded curves indicate the 25% & 75% percentile of the normalized errors among 560 test samples.

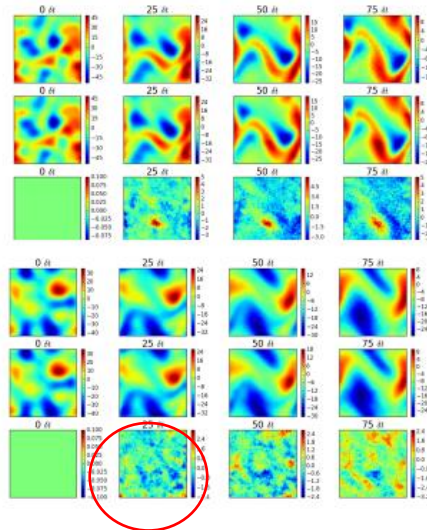


Figure 4. The first row shows the images of the true dynamics. The second row shows the images of the predicted dynamics using the PDE-Net having 3 δt -blocks with 5×5 (up) and 7×7 (down) filters. The third row shows the error maps. Time step $\delta t = 0.015$.

我们可以观察到如下结果：

(1) 即便使用有噪声的数据进行训练，PDE-net 仍然可以长久的预测动态。(figure 4 红圈部分)

(2) 多个 δt - Blocks 使得网络可以长久的预测动态。(figure 3, 随着从左到右 blocks 增加, 误差曲线下移)

(3) PDE-net 相比于冻结的 PDE-net 有显著优势, 尤其在 7×7 filters 时。(figure 3, 红色趋势线显然在蓝色趋势线之下; 这也是之前文中提到的可训练参数比冻结参数的优势体现。)

(4) PDE-net 在 7×7 filters 表现比 5×5 filters 有显著优势。(figure 4, 下方红色曲线比上方红色曲线显著靠下。)

识别隐藏的方程

对于线性问题, 可以通过 PDE 寻找系数 $\{c_{ij}: i + j \leq 4\}$ 来拟合 $\{f_{ij}: i + j \leq 4\}$ 。我们注意到

$f_{11} \cup \{f_{ij}: 2 \leq i + j \leq 4\}$ 在 (8) 式中没有出现, 而实际上 PDE-net 训练结果中, 这些项对应

的系数接近于 0。下面显示了 $i + j \leq 2$ 时的系数预测情况。

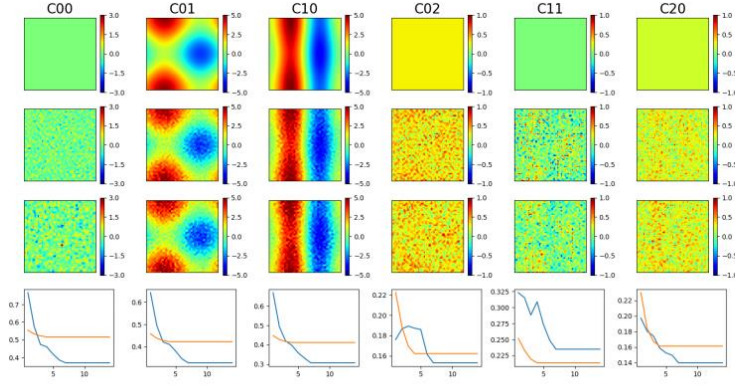


Figure 5. First row: the true coefficients of the equation. From the left to right are coefficients of u , u_x , u_y , u_{xx} , u_{xy} and u_{yy} . Second row: the learned coefficients by the PDE-Net with 6 δt -blocks and 5×5 filters. Third row: the learned coefficients by the PDE-Net with 6 δt -blocks and 7×7 filters. Last row: the errors between true and learned coefficients v.s. number of δt -blocks with different sizes of filters (5×5 blue and 7×7 orange).

我们可以发现，除了由于训练数据中加入的噪声导致的一些不确定性之外，网络学习到的系数 $\{c_{ij}\}$ 和 $\{f_{ij}\}$ 实际的系数非常接近。令人注意的是，将 filter 的大小从 5×5 提高到 7×7 对系数预测精度没有显著的改变。（figure 5 中的最后一行，红色曲线没有优势）

总结

本文中，我们设计了一个深度前馈网络，称为 PDE-net，来从动态演变中提取 PDE 模型，并继续对长期的演进进行预测。网络主要由两部分组成：

- （1）利用对 filters 进行适当的约束，使用卷积拟合微分算子。
- （2）利用传统机器学习方法或深度学习方法对非线性响应函数进行拟合。

PDE-net 可以对非常普通的、类似（1）形式的 PDE 进行预测。

$$u_t(t, x, y) = \mathbf{F}(x, y, u, u_x, u_y, u_{xx}, u_{xy}, u_{yy}, \dots), \quad (1)$$

可能的应用方向如下：

- （1）对数据中传感器无法直接测量的变量进行解释，提取其 PDE 模型；
- （2）对已有的 PDE 模型进行稳定的数值预测。（代替已有的数值方法）。

2.2 周四讨论结果

本周周四因为我与建伟均有课程没能与蔡老师见面，后确定昆捷交流改到每周周一。与此同时，蔡老师还发给我一份 `level-set` 的教程（主要讲其中的偏微分思想，约 500 页），让我对其进行学习。

我在上述 `PDE-net` 文章时时间较长，主要是本文涉及的数学概念较多，有的我花费了一些时间推导，有的我还在思考之中。

3. 下周工作计划

慢慢调整肝脏配准网络。

阅读 `PDE-net` 与 `Level Set- PDE` 的相关内容。

附表：工作整理

任务类型	任务内容	截止日期	当前进度
工作	肝脏分割比赛 (浙一举办) 负责 <code>registraion</code> 部分	结束	对肝脏配准继续进行研究、调整。
工作	神经纤维瘤研究 (中期目标)		蔡老师提出新方法：使用偏微分方程网络 <code>PDE-net</code> 对 <code>level set</code> 进行改进。正在学习相关内容。

本周工作时长：8 小时*4+ 2 小时*2 = 36 小时（中秋假期工作较少）。